Sains Malaysiana 38(2)(2009): 227–231

New Parallel R-point Explicit Block Method for Solving Linear High-Order Ordinary Differential Equations Directly

(Kaedah Baru R-titik Blok Selari Tak Tersirat bagi Menyelesaikan Persamaan Pembeza Biasa Linear Peringkat Tinggi Secara Langsung)

ZURNI OMAR & MOHAMED SULEIMAN

ABSTRACT

A new method called parallel R-point explicit block method for solving a single equation of higher order ordinary differential equation directly using a constant step size is developed. This method calculates the numerical solution at R point simultaneously and is parallel in nature. Computational advantages are presented by comparing the results obtained with the new method with that of the conventional 1-point method. The numerical results show that the new method reduces the total number of steps and execution time. The accuracy of the parallel block and the conventional 1-point method is comparable particularly when finer step sizes are used.

Keywords: Parallel R-point explicit block method; high-order ordinary differential equations

ABSTRAK

Satu kaedah baru R-titik blok selari tak tersirat bagi menyelesaikan persamaan pembeza biasa peringkat tinggi secara langsung dengan menggunakan saiz langkah malar dibangunkan. Kaedah selari ini menghitung penyelesaian berangka pada R titik serentak. Kelebihan pengiraan dipersembahkan dengan membandingkan keputusan yang diperolehi daripada kaedah baru dengan kaedah lazim 1-titik. Keputusan berangka menunjukkan kaedah baru mengurangkan jumlah bilangan langkah dan masa pengiraan. Ketepatan kaedah blok selari dan kaedah lazim 1-titik boleh dibandingkan terutamanya bila saiz langkah yang digunakan adalah kecil.

Kata kunci: Kaedah R-titik blok selari tak tersirat; persamaan pembeza biasa peringkat tinggi

INTRODUCTION

Consider the following linear dth order ODE in the subsequent discussion

$$y_d = f(x, y, y', y'', \ldots, y^{d-1}, y^{(i)}(a) = \eta_i, a \le x \le b. \quad (1)$$

Equation (1) can be solved by reducing it to the equivalent first order system and then solve it using first order ordinary differentials (ODEs) methods. The disadvantage of these methods is that the system in (1) has been enlarged. The other approach is to solve (1) directly as discussed in Gear (1966, 1971, 1978), Hall and Suleiman (1981) and Suleiman (1989). There have been quite a number of parallel methods for solving first order ODEs such as parallel block predictor-corrector method (Birta & Abou-Rabia 1987), multi-block methods (Chu & Hamilton 1987) and block implicit one-step methods (Shampine & Watts 1969).

A new parallel method called R-point explicit block method for solving linear high-order ODEs is proposed. This method is the extension of work in Omar and Suleiman (1999) and Omar et al. (2002) where the interval $[a,b]$ is divided into series of blocks with each block containing R points. For example, the points $x_{n-R+1}, x_{n-R+2}, \ldots, x_n$ in the first block while $x_{n+1}, x_{n+2}, \ldots, x_{n+R}$ in the second block (Figure 1) where solutions (1) are to be computed.

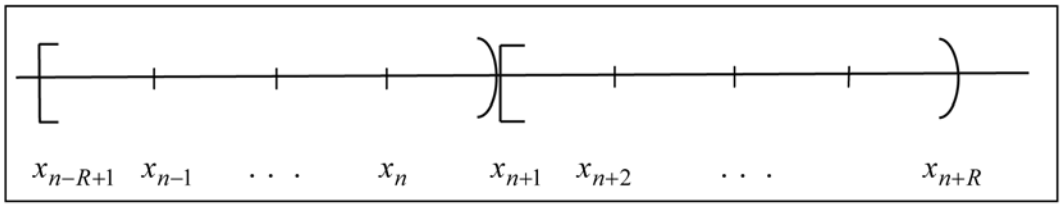

FIGURE 1. R-Point method

The computation which proceeds in blocks is based on the computed values at the earlier blocks. If the computed values at the previous k blocks are used to compute the current block containing R points, then the method is called R-point k-block method. Within a block it is possible to assign the computational tasks at each point to a single processor and therefore the computations can be performed simultaneously.

METHODS

DERIVATION OF THE PARALLEL R-POINT EXPLICIT BLOCK METHOD

Integrating Equation (1.1) p times gives

$$\int_{x_n}^{x_{n+1}}\int_{x_n}^{x}\int_{x_n}^{x}\int_{x_n}^{x}\ldots\int_{x_n}^{x} y^{d}\left(x,y,y',\ldots y^{d-1}\right)dx\ldots dx=\int_{x_n}^{x_{n+1}}\int_{x_n}^{x}\int_{x_n}^{x}\int_{x_n}^{x}\ldots\int_{x_n}^{x} f\left(x,y,y',\ldots y^{d-1}\right)dx\ldots dx \tag{2}$$

where

$$x_{n+j}=x_n+jh,\ j=1,2,3\ldots,R.$$

Define $P_{k,n}(x)$ as the interpolation polynomial which interpolates $f(x, y, y', \ldots, y^{d-1})$ at the k back values namely $\{x_{n-i} \mid i = 0, 1, 2, \ldots, k-1\}$ as follows

$$P_{k,n}(x)=\sum_{m=0}^{k-1}(-1)^m\binom{-s}{m}\nabla^m f_n$$

where

$$S=\frac{x-x_n}{h}.$$

Replacing with in (2.1) leads to

$$\begin{bmatrix} y_{n+1}^{(n-p)}\\ y_{n+2}^{(n-p)}\\ y_{n+3}^{(n-p)}\\ .\\ .\\ .\\ y_{n+R}^{(\ )}\end{bmatrix}=\begin{bmatrix} y_n^{(n-p)}\\ y_n^{(n-p)}\\ y_n^{(n-p)}\\ .\\ .\\ .\\ y_{n+R}^{(n-p)}\end{bmatrix}+h\begin{bmatrix} y_n^{(n-p+1)}\\ 2y_n^{(n-p+1)}\\ 3y_n^{(n-p+2)}\\ .\\ .\\ .\\ Ry_{n+R}^{(n-p)}\end{bmatrix}+\frac{h^2}{2!}\begin{bmatrix} y_n^{(n-p+2)}\\ 2^2y_n^{(n-p+2)}\\ 3^2y_n^{(n-p+2)}\\ .\\ .\\ .\\ R^2y_{n+R}^{(n-p)}\end{bmatrix}+\ldots\frac{h^{p-1}}{(p-1)!}\begin{bmatrix} y_n^{(n-1)}\\ 2^{p-1}y_n^{(n-1)}\\ 3^{p-1}y_n^{(n-1)}\\ .\\ .\\ .\\ R^{p-1}y_n^{(n-1)}\end{bmatrix}+\begin{bmatrix} A_1\\ A_2\\ A_3\\ .\\ .\\ .\\ A_R\end{bmatrix} \tag{3}$$

where

$$A_j=\int_{x_n}^{x_{n+j}}\int_{x_n}^{x}\int_{x_n}^{x}\int_{x_n}^{x}\ldots\int_{x_n}^{x}\sum_{m=0}^{k-1}(-1)^m\binom{-s}{m}\nabla^m f_n dx\ldots dx.$$

Substituting $dx = hds$ and changing the limit of integration in (3) leads to

$$\begin{bmatrix} y_{n+1}^{(n-p)}\\ y_{n+2}^{(n-p)}\\ y_{n+3}^{(n-p)}\\ .\\ .\\ .\\ y_{n+r}^{(n-p)}\end{bmatrix}=\begin{bmatrix} y_n^{(n-p)}\\ y_n^{(n-p)}\\ y_n^{(n-p)}\\ .\\ .\\ .\\ y_n^{(n-p)}\end{bmatrix}+h\begin{bmatrix} y_n^{(n-p+1)}\\ 2y_n^{(n-p+1)}\\ 3y_n^{(n-p+1)}\\ .\\ .\\ .\\ Ry_n^{(n-p+1)}\end{bmatrix}+\frac{h^2}{2!}\begin{bmatrix} y_n^{(n-p+2)}\\ 2^2y_n^{(n-p+2)}\\ 3^2y_n^{(n-p+2)}\\ .\\ .\\ .\\ R^2y_n^{(n-p+2)}\end{bmatrix}+\ldots$$

$$+\frac{h^{p-1}}{(p-1)!}\begin{bmatrix} y_n^{(n-1)}\\ 2^{p-1}y_n^{(n-1)}\\ 3^{p-1}y_n^{(n-1)}\\ .\\ .\\ .\\ R^{p-1}y_n^{(n-1)}\end{bmatrix}+h^p\begin{bmatrix} \sum_{m=0}^{k-1}\alpha_{1,m}^{(p)}\nabla^m f_n\\ \sum_{m=0}^{k-1}\alpha_{2,m}^{(p)}\nabla^m f_n\\ \sum_{m=0}^{k-1}\alpha_{3,m}^{(p)}\nabla^m f_n\\ .\\ .\\ .\\ \sum_{m=0}^{k-1}\alpha_{R,m}^{(p)}\nabla^m f_n\end{bmatrix} \tag{4}$$

where

$$\alpha_m^{(j,p)}=(-1)^m\int_0^j\frac{(1-s)}{(p-1)!}^{p-1}\binom{-s}{m}ds.$$

Define the generating functions as follows

$$G_j^{(p)}(t)=\sum_{m=0}^{\infty}\alpha_m^{(j,p)}t^m. \tag{5}$$

Solving (5) leads to the following relationships

$$G_j^{(p)}(t)=\frac{j^{p-1}-(p-1)!G_j^{(p-1)}(t)}{(p-1)!\log(1-t)}\ \text{for}\ p=2,3,\ldots,d \tag{6}$$

which can easily be verified using mathematical induction.

The solution of (6) is given by:

$$\begin{aligned}
&\alpha_0^{(1,1)}=1,\\
&\alpha_{m+1}^{(1,1)}=1-\sum_{r=0}^{m}\frac{\alpha_r^{(1,1)}}{m+2-r},\\
&\alpha_0^{(1,p)}=\alpha_1^{(1,p-1)},\\
&\alpha_{m+1}^{(1,p)}=\alpha_{m+2}^{(1,p-1)}-\sum_{r=0}^{m}\frac{\alpha_r^{(1,p)}}{m+2-r},\ \text{for}\ p=2,3,\ldots,n,\ m=0,1,\ldots,k-1.\\
&\alpha_0^{(j,1)}=j,\\
&\alpha_{m+1}^{(j,1)}=\frac{(m+j+1)(m+j)(m+j-1)\ldots(m+3)}{(d-1)!}-\sum_{r=0}^{m}\frac{\alpha_r^{(j,1)}}{m+2-r},\\
&\alpha_0^{(j,p)}=\alpha_0^{(j,p-1)}\\
&\alpha_{m+1}^{(1,p)}=\alpha_{m+2}^{(j,p-1)}-\sum_{r=0}^{m}\frac{\alpha_r^{(j,p)}}{m+2-r}\ \text{for j}=2,\ldots\text{R},\ p=2,3,\ldots,d,\ m=0,1,\ldots,k-1.
\end{aligned} \tag{7}$$

Note that formula (3) can be written in the form

$$\begin{bmatrix} y_{n+1}^{(n-p)}\\ y_{n+2}^{(n-p)}\\ y_{n+3}^{(n-p)}\\ .\\ .\\ .\\ y_{n+R}^{(n-p)}\end{bmatrix}=\begin{bmatrix} y_n^{(n-p)}\\ y_n^{(n-p)}\\ y_n^{(n-p)}\\ .\\ .\\ .\\ y_n^{(n-p)}\end{bmatrix}+h\begin{bmatrix} y_n^{(n-p+1)}\\ 2y_n^{(n-p+1)}\\ 3y_n^{(n-p+1)}\\ .\\ .\\ .\\ Ry_n^{(n-p+1)}\end{bmatrix}+\frac{h^2}{2!}\begin{bmatrix} y_n^{(n-p+2)}\\ 2^2y_n^{(n-p+2)}\\ 3^2y_n^{(n-p+2)}\\ .\\ .\\ .\\ R^2y_n^{(n-p+2)}\end{bmatrix}+\ldots \tag{8}$$

$$+\frac{h^{p-1}}{(p-1)!}\begin{bmatrix} y_n^{(n-1)} \\ 2^{p-1}\, y_n^{(n-1)} \\ 3^{p}\, y_n^{(n-1)} \\ . \\ . \\ . \\ R^{p-1} y_n^{(n-1)} \end{bmatrix} + h^p \begin{bmatrix} \sum_{m=0}^{k-1} \beta_{k-1,m}^{(1,p)} f_{-nm} \\ \sum_{m=0}^{k-1} \beta_{k-1,m}^{(2,p)} f_{n-m} \\ \sum_{m=0}^{k-1} \beta_{k-1,m}^{(3,p)} f_{n-m} \\ . \\ . \\ . \\ \sum_{m=0}^{k-1} \beta_{k-1,m}^{(R,p)} f_{n-m} \end{bmatrix}$$

where

$$\beta_{k-1,m}^{(j,p)} = (-1)^m \sum_{r=m}^{k-1} \binom{r}{m} \alpha_r^{(j,p)}. \tag{9}$$

TEST PROBLEMS

The following problems were tested on the Sequent Symmetry S27 using the 2-point and 3-point explicit block method.

Problem 1: $y''' = y + 3e^x, y(0) = 0, y'(0) = 1, y''(0) = 2,$
$0 \le x \le 1$
Solution: $y(x) = xe^x$
Source: Krogh (1968).

Problem 2: $y''' = 8y' = 3y - 4ex, y'(0) = -2, y''(0) = 10,$
$0 \le x \le 1$
Solution: $y(x) = e^x + e^{-3x}$
Source: Suleiman (1989).

Problem 3: $y^{(iv)} = (x^4 + 14x^3 + 49x^2 + 32x - 12)e^x,$
$y(0) = y'(0) = 0, y''(0) = 2, y^m(0) = -6, 0 \le x \le 1.$
Solution: $y(x) = x^2(1 - x)^2 e^x.$
Source: Russel and Shampine (1972).

RESULT AND DISCUSSIONS

NUMERICAL RESULTS

The numerical tests were performed on the shared memory parallel computer, Sequent S27 which has 6 processors. The programs for Explicit 1-Point (E1P) and the sequential implementation of the 2-Point Explicit Block (2PEB) and 3-Point Explicit Block (3PEB) methods were written in C language whereas parallel C language was used for the parallel implementation. Both languages were supported by the Sequent C library. Each method used 5 back values in its computation. The abbreviations and notations are defined as follows:

h Step size used
STEPS Total number of steps taken to obtain the solution
MTD Method employed
MAXE Magnitude of the maximum error of the computed solution
TIME The execution time in microseconds needed to complete the integration in a given range using the parallel computer Sequent S27.

S2PEB Sequential implementation of the 2-point explicit block method
P2PEB Parallel implementation of the 2-point explicit block method

S3PEB Sequential implementation of the 3-point explicit block method
P3PEB Parallel implementation of the 3-point explicit block method

The maximum error is defined as follows:

$$\text{MAXE} = \max_{1 \le i \le \text{STEPS}} \left(\left| y_i - y(x_i) \right| \right).$$

The comparison of the 2PEB and 3PEB methods with the E1P method for solving the test problems in terms of the total number of steps, maximum error and execution times are tabulated in Tables 1-3. Table 4 shows the ratio of steps and times of the 2PEB and 3PEB methods to E1P method. The ratios of the two parameters are obtained by dividing the parameters of the latter method with the corresponding parameters of the former methods. Hence, the ratios (also known as speedup) that are greater than one

TABLE 1. Comparison between the E1P, 2PEB and 3PEB methods for solving problem 1

h	MTD	STEPS	MAXE	TIME
	E1P	100	7.63447(-3)	121066
	S2PEB	53	7.63448(-3)	106065
10–2	P2PEB	53	7.63448(-3)	241380
	S3PEB	37	7.63449(-3)	115350
	P3PEB	37	7.63449(-3)	253305
	E1P	1000	7.62628(-4)	1126827
	S2PEB	503	7.62628(-4)	951148
10–3	P2PEB	503	7.62628(-4)	941787
	S3PEB	337	7.62628(-4)	1005923
	P3PEB	337	7.62628(-4)	833978
	E1P	10000	7.62546(-5)	11270476
	S2PEB	5003	7.62546(-5)	9486304
10–4	P2PEB	5003	7.62546(-5)	9129636
	S3PEB	3337	7.62546(-5)	9997548
	P3PEB	3337	7.62546(-5)	7963995
	E1P	100000	7.62539(-6)	112391084
	S2PEB	50003	7.62538(-6)	94637449
10–5	P2PEB	50003	7.62538(-6)	90659201
	S3PEB	33337	7.62538(-6)	99645969
	P3PEB	33337	7.62538(-6)	78263075

TABLE 2. Comparison between the E1P, 2PEB and 3PEB methods for molving problem 2

h	MTD	STEPS	MAXE	TIME
	E1P	100	1.18234(-1)	130511
	S2PEB	53	1.16165(-1)	135853
10–2	P2PEB	53	1.16165(-1)	256096
	S3PEB	37	1.10115(-1)	135708
	P3PEB	37	1.10115(-1)	254185
	E1P	1000	1.17139(-2)	1223205
	S2PEB	503	1.17115(-2)	1240065
10^{-3}	P2PEB	503	1.17115(-2)	1034676
	S3PEB	337	1.17042(-2)	1209884
	P3PEB	337	1.17042(-2)	913258
	E1P	10000	1.17030(-3)	12235538
	S2PEB	5003	1.17029(-3)	12371365
10^{-4}	P2PEB	5003	1.17029(-3)	10051687
	S3PEB	3337	1.17029(-3)	12036763
	P3PEB	3337	1.17029(-3)	8798665
	E1P	100000	1.17019(-4)	122141796
	S2PEB	50003	1.17019(-4)	123347394
10–5	P2PEB	50003	1.17019(-4)	99941712
	S3PEB	33337	1.17019(-4)	120063725
	P3PEB	33337	1.17019(-4)	87246831

TABLE 3. Comparison between the E1P, 2PEB and 3PEB methods for solving problem 3

h	MTD	STEPS	MAXE	TIME
	E1P	100	1.00778(-2)	210474
	S2PEB	53	1.00778(-2)	222259
10–2	P2PEB	53	1.00778(-2)	296038
	S3PEB	37	1.00779(-2)	201983
	P3PEB	37	1.00779(-2)	325724
	E1P	1000	1.00078(-3)	2006247
	S2PEB	503	1.00078(-3)	2076058
10–3	P2PEB	503	1.00078(-3)	1623001
	S3PEB	337	1.00078(-3)	1850179
	P3PEB	337	1.00078(-3)	1583939
	E1P	10000	1.00008(-4)	20077275
	S2PEB	5003	1.00008(-4)	20727420
10–4	P2PEB	5003	1.00008(-4)	15922113
	S3PEB	3337	1.00008(-4)	18504213
	P3PEB	3337	1.00008(-4)	15212065
	E1P	100000	1.00001(-5)	200307659
	S2PEB	50003	1.00001(-5)	206611648
10–5	P2PEB	50003	1.00001(-5)	158493054
	S3PEB	33337	1.00001(-5)	184047416
	P3PEB	33337	1.00001(-5)	151403794

TABLE 4. The Ratio steps and execution times of 2PEB and 3PEB methods to the E1P method for solving higher order ODEs

TOL	MTD	RATIO STEP	RATIO PROB.1	TIME PROB.2	PROB.3
	S2PEB	1.88679	1.14143	0.96068	0.94698
	P2PEB	1.88679	0.50156	0.50962	0.71097
10–2	S3PEB	2.70270	1.04955	0.96171	1.04204
	P3PEB	2.70270	0.47795	0.51345	0.64617
	S2PEB	1.98807	1.18470	0.98640	0.96637
	P2PEB	1.98807	1.19648	1.18221	1.23613
10–3	S3PEB	2.96736	1.12019	1.01101	1.08435
	P3PEB	2.96736	1.35115	1.33939	1.26662
	S2PEB	1.99880	1.18808	0.98902	0.96863
	P2PEB	1.99880	1.23449	1.21726	1.26097
10–4	S3PEB	2.99670	1.12732	1.01651	1.08501
	P3PEB	2.99670	1.41518	1.39061	1.31983
	S2PEB	1.99988	1.18760	0.99023	0.96949
	P2PEB	1.99988	1.23971	1.22213	1.26383
10–5	S3PEB	2.99967	1.12790	1.01731	1.08835
	P3PEB	2.99967	1.43607	1.39996	1.32300

for both parameters indicate the efficiency of the 2PEB and 3PEB methods.

CONCLUSION

It is apparent from the results that the 3PEB method outperforms the E1P method in terms of the total number of steps. As the step size becomes finer, the 3PEB method reduces the number of steps to almost one half. These results are expected since the 3PEB method approximates the numerical solution at three points respectively at the same time, thus reducing the number of steps taken by the method. In term of accuracy, all methods have the same order of accuracy. The execution times taken by the parallel implementation of the 2PEB and 3PEB methods are more than those taken by the sequential counterparts and the E1P method at $h = 10^{-2}$. This is because the number of steps taken is small and most of the execution times are dominated by the parallel overheads. However, Table 1-4 show that the timings of the parallel version of the 2PEB and 3PEB methods are better then other methods when $h \leq 10^{-3}$. The reason for these gains is that as the step size gets smaller, more steps are taken to complete the computation. By using 2 and 3 processors instead of 1, the computation can be performed quicker. In other words, the parallelism in the 2PEB and 3PEB methods could really be exploited. The results also suggest that parallel 3PEB method is recommended for solving second order ODEs directly using finer step sizes.

REFERENCES

Birta L.G. & Abou-Rabia O. 1987. Parallel Block Predictor-Corrector Methods for ODEs, *IEEE Transactions on Computers* C-36(3): 299-311.

Chu M.T. & Hamilton H. 1987. Parallel Solution of ODEs by Multi-Block Methods, *Siam J. Sci. Stat. Comput*.8(1): 342-353.

Gear, C.W. 1966. The Numerical Integration of Ordinary Differential Equations. *Math. Comp*. 21: 146-156.

Gear, C.W. 1971. *Numerical Initial Value Problems in Ordinary Differential Equations*. New Jersey: Prentice Hall, Inc.

Gear, C.W. 1978. The Stability of Numerical Methods for Second-Order Ordinary Differential Equations. *SIAM J. Numer. Anal*. 15(1): 118-197.

Hall, G. & Suleiman, M.B. 1981. Stability of Adams-Type Formulae for Second-Order Ordinary Differential Equations. *IMA J. Numer. Anal*. 1: 427-428.

Krogh, F.T. 1968. A Variable Step, Variable Order Multistep Method for the Numerical Solution of Ordinary Differential Equation. Proceedings of the IFIP Congress in Information Processing 68: 194-199.

Omar, Z.B. & Suleiman, M.B. 1999. New Parallel 3-Point Explicit Block Method for Solving Second Order Ordinary Differential Equations (ODEs) Directly**.** *Jurnal ANALISIS* 6(1&2): 61-74.

Omar, Z.B., Suleiman, M.B., Saman, M.Y. & Evans, D.J. 2002. Parallel R-point Explicit Block Method for Solving Second Order Ordinary Differential Equations Directly. *International Journal of Computer Mathematics* 9: 289-298.

Russel, R.D. & Shampine, L.F. 1972. A Collocation Method for Boundary Value Problems. *Num. Math* 19: 1-28.

Shampine L.F. & Watts H.A. 1969. Block Implicit One-step Methods*, Math. Comp*. 23: 731-740.

Suleiman, M.B. 1989. Generalised Muiltistep Adams and Backward Differentiation Methods for the Solution of Stiff and Non-Stiff Ordinary Differential Equations. PhD Thesis. University of Manchester.

Suleiman, M.B. 1989. Solving Higher Order ODEs Directly by the Direct Integration Method. *Applied Mathematics and Computation* 33(3): 197-219.

Zurni bin Omar
College of Arts and Sciences
Universiti Utara Malaysia
06010 UUM, Sintok, Kedah
MALAYSIA

Mohamed bin Suleiman
Department of Mathematics
Universiti Putra Malaysia
43400 Serdang, Selangor D.E.
MALAYSIA

Received: 23 April 2008
Accepted: 8 August 2008